

APPENDIX

Condition Codes	Description	Flags
0 0 0 0	Unconditional	Don't care
0 0 0 1	Lower than, Carry	C
0 0 1 0	Lower than or same	C+Z
0 0 1 1	Higher than	~C, ~Z
0 1 0 0	Higher than or same, No Carry	~C
0 1 0 1	Equal, Zero	Z
0 1 1 0	Not equal, Not zero	~Z
0 1 1 1	Less than	(N, ~V) + (~N, V)
1 0 0 0	Less than or equal	(N, ~V) + (~N, V) + Z
1 0 0 1	Greater than	(N, V, ~Z) + (~N, ~V, Z)
1 0 1 0	Greater than or equal	(N, V) + (~N, ~V)
1 0 1 1	Positive	~N, ~Z
1 1 0 0	Negative	N
1 1 0 1	Non-negative	~N
1 1 1 0	Overflow	V
1 1 1 1	No overflow	~V

CONDITION CODES SUMMARY

scodes/dcodes	P0:MFLAGS	C0:LC1	L0:LE1
	P1:OPTIONS	C1:LC2	L1:LE2
	P2:SYNC/PP#	C2:LC3	L2:LE3
000:D0-D7	P3:INTEN	C3:RC1	L3:LS
001:P0-P7	P4:INTFLG	C4:RC2	L4:reserved
010:A0-A7	P5:SR	C5:RC3	L5:reserved
011:X0-X7	P6:RET	C6:reserved	L6:reserved
	P7:PC	C7:reserved	L7:reserved

REGISTER CODES SUMMARY

2 bits	4 bits		
0 0 = *	0 0 0 0 = 3op	1 0 0 0 = pushl/popl	
0 1 = X+m	0 0 0 1 = X+m	1 0 0 1 = I+m	
1 0 = -Im	0 0 1 0 = PushRet	1 0 1 0 = move	
1 1 = I+m	0 0 1 1 = X-m	1 0 1 1 = I-m	
	0 1 0 0 = +X	1 1 0 0 = +I	
	0 1 0 1 = +Xm	1 1 0 1 = +Im	
	0 1 1 0 = -X	1 1 1 0 = -I	
	0 1 1 1 = -Xm	1 1 1 1 = -Im	

ADDRESSING MODE CODES SUMMARY

Absolute

ABS

Syntax ABS src,dst

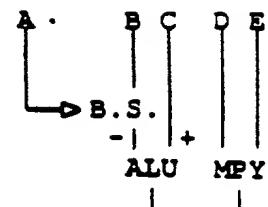
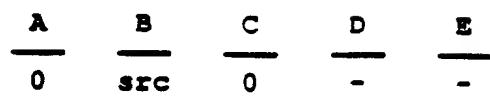
Operation $|src| \rightarrow dst$

Operands

D,D With parallel transfers

any,any No parallel transfers

Routing



Encoding	31	27	22	19	16	11	8	5	0
Parallel transfers									
	0 0 0 0	src	dst					
	0 0 0 0	src	dst	0 0 0 0 0	scde	dcde	0 0 0	0 0 0

Description The absolute value of src is loaded into dst.

Status Bits N - Set to 0.

C - Unaffected.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Add

ADD

Syntax ADD src1,src2,dst

Operation src1 + src2 → dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

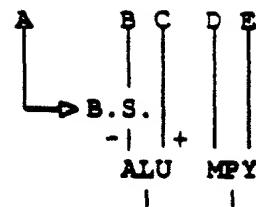
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
—	—	—	—	—
0	src2	src1	-	-



Encoding

31	30	27	22	19	16	15	11	8	5	2	0
Parallel transfers											
1	src2	src1	dst						
1	src2	src1	dst	0 0 0 0 0	scd1	dcda	scde2	0 0 0	
0 1 1 1	src1	dst	H					16 bit immediate
0 0 1 1	code	dst	H					16 bit immediate

Description The sum of src1 and src2 is loaded into dst. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a carry occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Add with Carry

ADDC

Syntax ADDC *src1,src2,dst*

Operation $src1 + src2 + C \rightarrow dst$

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

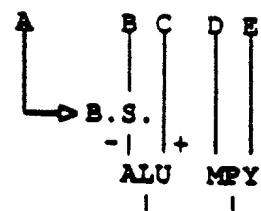
D,any,any No parallel transfers

D,I,D No parallel transfers

any,L,src1 No parallel transfers

Routing

A	B	C	D	E
—	—	—	—	—
0	src2	src1	-	-



Encoding

3130		27	22	19	1615	11	8	5	2	0
Parallel transfers										
1	src2	src1	dst	0 0 0 0 0	scde1	dcde	scde2	0 0 0	
0 1 1 1	src1	dst	H	16 bit immediate					
0 0 1 1	code	dst	H	16 bit immediate					

Description The sum of *src1* and *src2* and the Carry flag is loaded into *dst*. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a carry occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Add Multiple byte half-words

ADDM

Syntax ADDM *src1 src2 dst*

Operation *src1* + + + + / + + / + *src2* → *dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

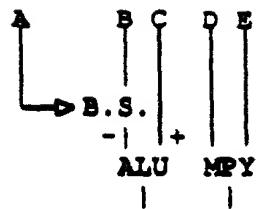
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,*src1* No parallel transfers

Routing

A	B	C	D	E
—	src2	src1	—	—



	31	30	27	22	19	16	15	11	8	5	2	0					
1	src2	.	.	.	src1	dst	Parallel transfers										
1	src2	.	.	.	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0
0	1	1	1	.	.	.	src1	dst	H	16 bit immediate							
0	0	1	1	.	.	.	code	dst	H	16 bit immediate							

Description The ALU bits in the OPTIONS register are used to split the ALU into four separate bytes, two separate half-words, or one word. The sum of each individual portion of *src1* and *src2* is loaded into *dst*. The individual carries are stored into the 4, 2 or 1 least-significant M bits respectively. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits The carries from the individual adds are stored in the least-significant 4, 2 or 1 M bits, according to the ALU option bit values.

Examples

AND

AND

Syntax AND src1,src2,dst

Operation src1 AND src2 → dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

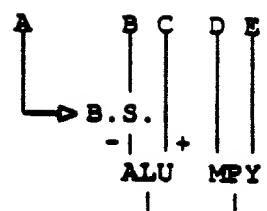
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Rounding

A	B	C	D	E
—	—	—	—	—
0	src2	src1	-	-



Encoding 3130 27 22 19 1615 11 8 5 2 0

Parallel transfers									
1	src2	...	src1	dst	0	0	0	0	0
1	src2	...	src1	dst	0	0	0	0	scde1 dcde scde2 0 0 0
0 1 1 1	...	src1	dst	H	16 bit immediate				
0 0 1 1	...	code	dst	H	16 bit immediate				

Description The AND of src1 and src2 is loaded into dst. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

AND with Complement

ANDN

Syntax ANDN src1,src2,dst

Operation src1 AND (NOT src2) → dst

Operands

D,D,D With parallel transfers

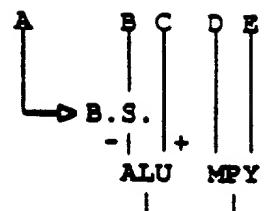
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing



Encoding

31	30	27	22	19	16	15	11	8	5	2	0						
1	src2	...	src1	dst	Parallel transfers												
1	src2	...	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0		
0	1	1	1	...	src1	dst	H	16 bit immediate									
0	0	1	1	...	code	dst	H	16 bit immediate									

Description The AND of src1 with the 1s complement of src2 is loaded into dst. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Conditional Branch relative to PC

BRcond

Syntax BRcond 24 bit immediate

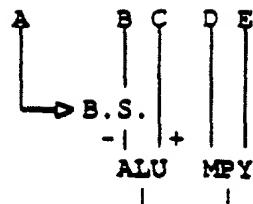
Operation 24 bit Immediate + PC conditionally loaded into PC

Operands

I No parallel transfers

Routing

A	B	C	D	E
0	Imm	PC	-	-



Encoding 31 27 23

0 0 1 0	cond.	24 bit immediate	0
---------	-------	------------------	---

Description The 24 bit immediate is added to the 24 bit Program Counter, and conditionally returned to the PC.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - Unaffected

M Bits Unaffected

Examples

Clear Bit

CLRB

Syntax CLRB src1,src2,dst

Operation src1[src2]:=0, src1' → dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

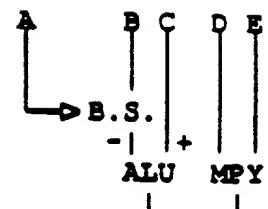
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

<u>A</u>	<u>B</u>	<u>C</u>	<u>D</u>	<u>E</u>
src2	1	src1	-	-



3130 27 22 19 1615 11 8 5 2 0															
1	src2	src1	dst	Parallel transfers										
1	src2	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0
0	1	1	1	src1	dst	H	16 bit immediate							
0	0	1	1	code	dst	H	16 bit immediate							

Description The bit within src1 pointed to by the least-significant 5-bits of src2 is set to zero. The result is loaded into dst. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Compare

CMP

Syntax CMP *src1 src2*

Operation *src1 - src2*, status bits set on result.

Operands

D,D With parallel transfers

any,D No parallel transfers

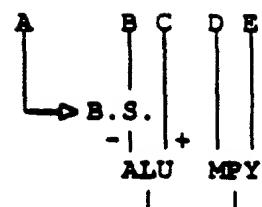
D,any No parallel transfers

D,I No parallel transfers

any,I No parallel transfers

Routing

A	B	C	D	E
—	—	—	—	—
0	<i>src2</i>	<i>src1</i>	—	—



Encoding

	3130	27	22	19	1615	11	8	5	2	0
Parallel transfers										
1	<i>src2</i>	<i>src1</i>	0 0 0					
1	<i>src2</i>	<i>src1</i>	0 0 0	0 0 0 0 0	<i>scde1</i>	0 0 0	<i>scde2</i>	0 0 0
0 1 1 1	<i>src1</i>	0 0 0	H				
0 0 1 1	code	<i>src1</i>	H					

Description *src2* is subtracted from *src1*, but the result is not loaded into any destination. The status bits are set according to the result. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a carry occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Compare Multiple bytes/half-words

CMPM

Syntax CMPM *src1 src2*

Operation *src1* $\sim\sim\sim\sim\sim\sim$ *src2*, M bits set on result(s).

Operands

D,D With parallel transfers

any,D No parallel transfers

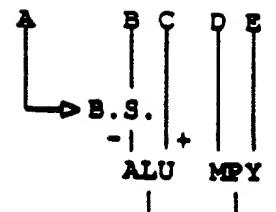
D,any No parallel transfers

D,I No parallel transfers

any,I No parallel transfers

Routing

A	B	C	D	E
0	<i>src2</i>	<i>src1</i>	-	-



Encoding	31	30	27	22	19	16	15	11	8	5	2	0
Parallel transfers												
1	<i>src2</i>	.	.	.	<i>src1</i>	0	0	0				
1	<i>src2</i>	.	.	.	<i>src1</i>	0	0	0	0	0	0	0
0 1 1 1	<i>src1</i>	0	0	0	H			
0 0 1 1	code	<i>src1</i>	H					

Description The ALU bits in OPTIONS are used to split the ALU into four separate bytes, two separate half-words, or one word. The portions of *src2* are subtracted from *src1* and then each result is compared with zero. The individual zero bits are stored into the 4, 2 or 1 least-significant M bits respectively. Carry flag is loaded into *dst*. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits The zero bits of the individual compares are stored into the 4, 2 or 1 least-significant M bits, according to the ALU option bits.

Examples

Divide Iteration

DIVI

Syntax DIVI src1,src2,(src1)

Operation If $src1 - src2 < 0$, then $src1 := src1 \times 2$, else $src1 := ((src1 - src2) \times 2) + 1$

Operands

D,D,src1 With parallel transfers

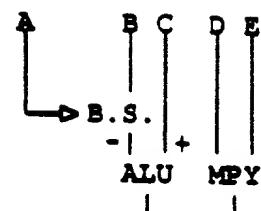
any,D,src1 No parallel transfers

D,any,src1 No parallel transfers

D,I,src1 No parallel transfers

any,I,src1 No parallel transfers

Routing



	31	30	27	22	19	16	15	11	8	5	2	0
1	src2	.	.	.	src1	0 0 0						Parallel transfers
1	src2	.	.	.	src1	0 0 0	0 0 0 0 0 0	scde1	0 0 0	scde2	0 0 0	
0 1 1 1	src1	0 0 0	H					16 bit immediate
0 0 1 1	code	src1	H					16 bit immediate

Description $src2$ is subtracted from $src1$. If the result is negative then $src1$ is left-shifted by one bit, and a zero is inserted into bit 0. This is then loaded into $src1$. If however the result is zero or positive then the result of the subtraction is left-shifted one bit and a 1 is inserted into bit 0. This is then loaded into $src1$. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - Unaffected

M Bits Unaffected

Examples

Idle until interrupt

IDLE

Syntax IDLE

Operation *wait for an enabled interrupt*

Operands

None With parallel transfers

Encoding	31	27	22	19	16	0
	0 0 0 0	0 0 0	0 0 0	Parallel transfers	

Description Instruction waits until an enabled interrupt occurs before proceeding. If parallel transfers are coded then they will happen after the interrupt has occurred, but before the interrupt routine is executed.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Lock MIMD PPs together

LCK

Syntax LCK

Operation wait for MIMD PPs to synchronise

Operands

None With parallel transfers

Encoding 31 27 22 19 16 0

0	0	0	0	0	0	0	0	0	0	Parallel transfers
---	---	---	---	-------	---	---	---	---	---	---	--------------------

Description This instruction is used to begin a piece of MIMD synchronised PP code. It will cause the PP to wait until all the PPs indicated by 1s in the SYNC register are in sync with each other. The following instructions will then be fetched in-step with the other MIMD PPs. (Execution of the Address and Execute pipeline stages will occur as each successive instruction is synchronously fetched). ULCK will terminate synchronous code execution.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional Load

LDcond

Syntax LDcond *An(mode).dst

Operation *src conditionally → dst

Operands

~~cond A0-A7(mode(g),<Dn,An,Xn,Pn>)~~ No parallel Data Unit operation.

Addressing modes pre- or post-indexing

+/- 3 bit immediate or +/- an index register

modify Address register, or leave unaltered

Encoding	31	27	232221	18	1615	1110	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 0 0 0	mod	0 cde	A	dst			Imm/X

Description This instruction will conditionally load a Dn, An, Xn or Pn register from an indirect address generated from any Address register. (Mn, Qn, Cn or Ln cannot be directly loaded from memory except with a POP instruction). If specifying an immediate, then +/- 0 to 7 are available. If specifying an index register, then it must be in the same Address sub-unit as the Address register. If the address is non-aligned, then this will load only the lower byte(s). Note that if a register modify is specified then this will happen unconditionally. Only the actual load is conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modify but without loading a register.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional Load Immediate

LDIcond

Syntax LDIcond 16-bit immediate.dst

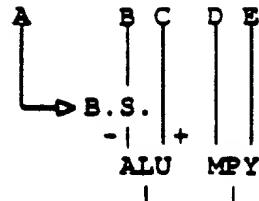
Operation 16-bit immediate conditionally loaded into dst

Operands

I,any No parallel transfers

Routing

A	B	C	D	E
0	src	0	-	-



Encoding 31 27 2322 19 1615 0

0	1	1	0	cond.	1	dcde	dst	S	16 bit immediate	0
---	---	---	---	-------	---	------	-----	---	------------------	---

Description The 17-bit signed immediate is loaded into dst only if cond is true. The sign bit is extended through to bit 31.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Load Loop Start or Program Counter

LDLS, LDPC

Syntax LD 24-bit immediate,ds:

Operation 24-bit immediate loaded into ds:

Operands

I,LS No parallel transfers

I,PC No parallel transfers

Encoding	31	23	0
	0 0 0 1 1 1 .		24 bit immediate

Description Load the 24-bit immediate into either the Loop Start address register, or the Program Counter.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional Load 1 Load

LD||LDcond

Syntax LDcond *An(mode).dst1 || LD *An(mode).dst2

Operation {I *src1 conditionally (s.) dst1, *src2 conditionally → dst2}

Operands cond A0-A3(mode(g),Dm A4-A7(mode),Dn) No parallel Data Unit operation.

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated indeX register with Address register modify

indirect without indexing

Encoding 31 27 232221 18 1615 1110 8 7 5 2 0

0	1	1	0	cond.	0	N	0	0	0	0	0	1	mdg	mdl	0	A47	0	A03	Dg	D1
---	---	---	---	-------	---	---	---	---	---	---	---	---	-----	-----	---	-----	---	-----	----	----

Description This instruction will conditionally load dst1 via the Global bus from an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will conditionally (same condition) load dst2 via the Local bus from an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independantly on the two buses. The indeX register(s) used have the same subscript(s) as the Address register(s). The dsts must be D registers. Note that if register modifies are specified then these will happen unconditionally. Only the actual loads are conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modifies but without loading registers.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional Load : store

LD¹STcond**Syntax** LDcond *Am(mode),dst1 || ST src2,*An(mode)**Operation** (I *src1 conditionally {s.} dst1, src2 conditionally {s.} *dst2)**Operands**

cond A0-A3(mode(g),Dm Dn,A4-A7(mode)) No parallel Data Unit operation.

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated indeX register with Address register modify

indirect without indexing

Encoding	31	27	232221	18	1615	1110	8	7	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 0 0 1	mdg	mdl	0 A47	1 A03	Dg	D1		

Description This instruction will conditionally load *dst1* via the Global bus from an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will conditionally (same condition) store *src2* via the Local bus to an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independantly on the two buses. The indeX register(s) used have the same subscript(s) as the Address register(s). *dst1* and *src2* must be D registers. Note that if register modifies are specified then these will happen unconditionally. Only the actual loads and stores are conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modifies but without loading or storing anything.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected**Examples**

Load Upper Cond. nally

LDUcond

Syntax LDUcond *An(mode),dst

Operation *src conditionally → dst

Operands

A0-A7(mode(g),Dn) No parallel Data Unit operation.

Addressing modes pre- or post-indexing

+/- 3 bit immediate or +/- an index register

modify Address register, or leave unaltered

Encoding	31	27	232221	18	1615	1110	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 1 0 0	. mod	0 cde	A	dst	Imm/X		

Description For use with non-aligned addresses. Conditionally loads the upper portion of a PP register from an indirect non-aligned address generated from any Address register. The destination PP register must be a D register. (Non-aligned loads are not supported to any other PP registers). If specifying an immediate then +/- 0 to 7 are available. If specifying an index register then it must be in the same Address sub-unit as the Address Note that if register modify is specified then this will happen unconditionally. Only the actual load is conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modify but without loading a register. register.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Left-most 1

LM1

Syntax LM1 src,dst

Operation bit number of left-most 1 in src → dst

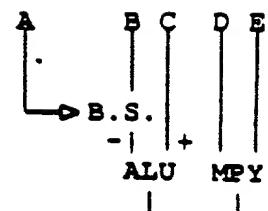
Operands

D,D With parallel transfers

any,any No parallel transfers

Routing

A	B	C	D	E
0	src	0	-	-



Encoding 31 27 22 19 16 11 8 5 0

0 0 0 0	src	dst	Parallel transfers				
0 0 0 0	src	dst	0 0 0 0 0	scde	dcde	0 0 0	0 0 0

Description The bit number of the left-most 1 in src is loaded into dst.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - 1 if src contained all zeros, 0 otherwise.

M Bits Unaffected

Examples

D to D Register M

MOV

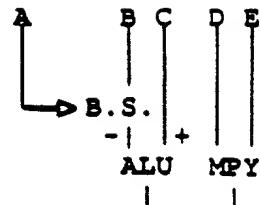
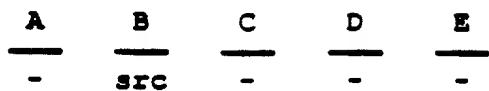
Syntax MOV src dst

Operation src → dst

Operands

D,D With parallel transfers

Routing



Encoding	31	27	22	19	16	0
	0 0 0 0	src	dst	Parallel transfers	

Description Any D register src is moved to any D register dst.

Status Bits N - 1 if src is negative, 0 otherwise

C - Unaffected.

V - Unaffected.

Z - 1 if src is zero, 0 otherwise

M Bits Unaffected

Examples

Conditional Move

MOVcond

Syntax MOVcond *src,dst*

Operation *src conditionally → dst*

Operands

cond any,any No parallel Data Unit operation.

<i>Encoding</i>	31	27	232221	18	16	11	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 0 0 0 1 0 1 0	scde	dcde	src	dst			

Description This instruction will conditionally move any register *src* to any other register *dst*.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Signed Multiply

MPY

Syntax MPY src1,src2,dst

Operation src1 \times src2 \rightarrow dst

Operands

D,D,D With parallel transfers

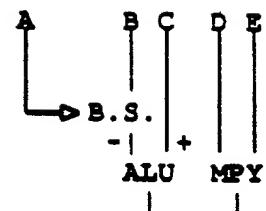
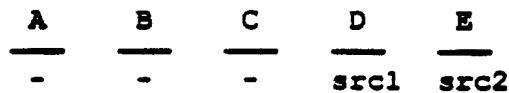
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing



Encoding 3130 27 22 19 16 11 8 5 2 0

Parallel transfers									
1	src2	src1	dst					
1	src2	src1	dst	0 0 0 0 0	scde1	dcde	scde2	0 0 0
0 1 1 1	src1	dst	H	16 bit immediate				
0 0 1 1	code	dst	H	16 bit immediate				

Description The 32-bit product of the signed 16 LS bits of src1 and the signed 16 LS bits of src2 is loaded into dst. The immediate is assumed to be a signed 16-bit quantity.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Unsigned Multiply

MPYU

Syntax MPYU src1,src2,dst

Operation $src1 \times src2 \rightarrow dst$

Operands

D,D,D With parallel transfers

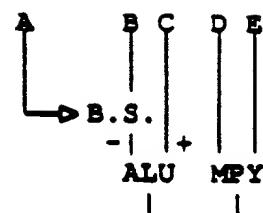
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing



Encoding	31	27	22	19	16	0	
1	src2	...	src1	dst	Parallel transfers		
1	src2	...	src1	dst	0 0 0 0 0	scde1	dcde
0 1 1 1	...	src1	dst	H	16 bit immediate		
0 0 1 1	...	code	dst	H	16 bit immediate		

Description The 32-bit product of the unsigned 16 LS bits of $src1$ and the unsigned 16 LS bits of $src2$ is loaded into dst . The immediate is assumed to be an unsigned 16-bit quantity.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Multiply with Paral. Add

MPY|ADD

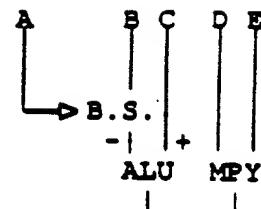
Syntax MPY src1.src2.dst1 || ADD src3.dst2

Operation $src1 \times src2 \rightarrow dst1, dst2 + src3 \rightarrow dst2$

Operands

D4-7,D0-3,D2-5,D0-3,D2-5 With parallel transfers

Routing



Encoding 31 2726 24 22 20 18 16

0

1	1	0	0	0	sc3	ds2	sc1	sc2	ds1	Parallel transfers
---	---	---	---	---	-----	-----	-----	-----	-----	--------------------

Description The 32-bit product of the signed 16 LS bits of src1 and the signed 16 LS bits of src2 is loaded into dst1. In parallel, the sum of src3 and dst2 is generated and loaded into dst2. Status bits are set according to the add result. The register range of the operands is subject to revision.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if carry occurs, 0 otherwise

V - 1 if overflow occurs, 0 otherwise

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Multiply with Parallel Shift and Add

MPY | SADD

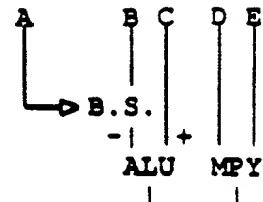
Syntax MPY src1,src2,dst1 || SADD src3,dst2

Operation $src1 \times src2 \rightarrow dst1$, $dst2 + (src3 \text{ shifted by predefined amount}) \rightarrow dst2$

Operands D4-7,D0-3,D2-5,D0-3,D2-5 With parallel transfers

Routing

A	B	C	D	E
pre	src3	dst2	src1	src2



Encoding 31 2726 24 22 20 18 16 0

1	1	0	1	1	sc3	ds2	sc1	sc2	ds1	Parallel transfers			
---	---	---	---	---	-----	-----	-----	-----	-----	--------------------	--	--	--

Description The 32-bit product of the signed 16 LS bits of src1 and the signed 16 LS bits of src2 is loaded into dst1. In parallel, src3, shifted by a predetermined shift amount contained in the OPTIONS register, is added to dst2. The result is loaded into dst2. Status bits are set according to the add result. The register range of the operands is subject to revision.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if carry occurs, 0 otherwise

V - 1 if overflow occurs, 0 otherwise

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Multiply with Parallel Shift and Subtract

MPY|SSUB

Syntax MPY src1,src2,ds1 || SSUB src3,ds2

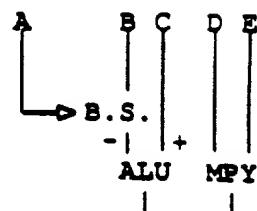
Operation $src1 \times src2 \rightarrow dst1$, $dst2 - (src3 \text{ shifted by predefined amount}) \rightarrow dst2$

Operands

D4-7,D0-3,D2-5,D0-3,D2-5 With parallel transfers

Routing

A	B	C	D	E
pre	src3	dst2	src1	src2



Encoding 31 2726 24 22 20 18 16 0

1	1	0	1	1	sc3	ds2	sc1	sc2	ds1	Parallel transfers		
---	---	---	---	---	-----	-----	-----	-----	-----	--------------------	--	--

Description The 32-bit product of the signed 16 LS bits of src1 and the signed 16 LS bits of src2 is loaded into dst1. In parallel, src3, shifted by a predetermined shift amount contained in the OPTIONS register, is subtracted from dst2. The result is loaded into dst2. Status bits are set according to the subtract result. The register range of the operands is subject to revision.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if carry occurs, 0 otherwise

V - 1 if overflow occurs, 0 otherwise

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Multiply with Par . Subtract

MPY|SUB

Syntax MPY src1,src2,ds1 || SUB src3,ds2

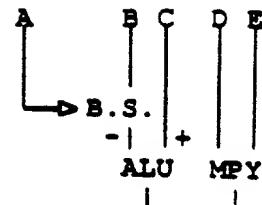
Operation src1 × src2 → dst1, dst2 - src3 → dst2

Operands

D4-7,D0-3,D2-5,D0-3,D2-5 With parallel transfers

Routing

A	B	C	D	E
0	src3	dst2	src1	src2



Encoding 31 2726 24 22 20 18 16

1	1	0	0	0	sc3	ds2	sc1	sc2	ds1	Parallel transfers			
0	0	0	0	0	0	0	0	0	0				

Description The 32-bit product of the signed 16 LS bits of src1 and the signed 16 LS bits of src2 is loaded into dst1. In parallel, the src3 is subtracted from dst2 and the result loaded into dst2. Status bits are set according to the subtract result. The register range of the operands is subject to revision.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if carry occurs, 0 otherwise

V - 1 if overflow occurs, 0 otherwise

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Merge Multiple bytes/half-words

MRGM

Syntax MRGM *src1,src2,dst*

Operation *src1 <merge> src2 → dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

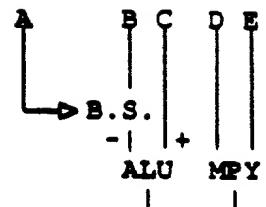
D,any,any No parallel transfers

D,I,D No parallel transfers

any,L,src1 No parallel transfers

Routing

A	B	C	D	E
0	src2	src1	-	-



Encoding 3130 27 22 19 1615 11 8 5 2 0

Parallel transfers											
1	src2	.	.	.	src1	dst					
1	src2	.	.	.	src1	dst	0 0 0 0 0	scde1	dcda	scde2 0 0 0	
0 1 1 1	src1	dst	H	16 bit immediate			
0 0 1 1	code	dst	H	16 bit immediate			

Description The ALU bits in the OPTIONS register are used to split the ALU into four separate bytes, two separate half-words, or one word. The 4, 2 or 1 least-significant M bits respectively are used to multiplex the individual portions of *src1* and *src2* into *dst*. Where an M bit is a 1 the portion of the result comes from *src2*, and where the M bit is a 0 the portion of the result comes from *src1*. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

NAND

NAND

Syntax NAND *src1,src2,dst*

Operation *src1 NAND src2 → dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

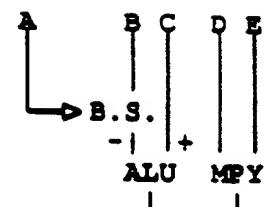
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
—	—	—	—	—
0	src2	src1	—	—



Encoding 3130 27 22 19 1615 11 8 5 2 0

1	src2	...	src1	dst	Parallel transfers										
1	src2	...	src1	dst	0	0	0	0	0	scdcl	dcde	scde2	0	0	0
0	1	1	1	...	src1	dst	H	16 bit immediate							
0	0	1	1	...	code	dst	H	16 bit immediate							

Description The NAND of *src1* and *src2* is loaded into *dst*. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Negate

NEG

Syntax NEG src dst

Operation 2s complement of src \rightarrow dst

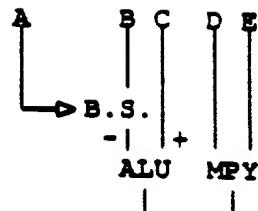
Operands

D,D With parallel transfers

any,any No parallel transfers

Routing

A	B	C	D	E
0	src	0	-	-



Encoding

31	27	22	19	16	11	8	5	2	0
0 0 0 0	src	dst	Parallel transfers					
0 0 0 0	src	dst	0 0 0 0 0	scde	dcde	0 0 0	0 0 0	

Description The 2s complement of src is loaded into dst.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a borrow occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Negate with Borrow

NEGB

Syntax **NEGB src,dst**

Operation ($2s$ complement of *src*) - $C \rightarrow dst$

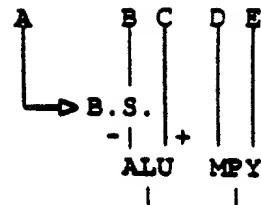
Operands

D,D With parallel transfers

any,any No parallel transfers

Routing

A	B	C	D	E
0	src	0	-	-



Encoding 31 27 22 19 16 11 8 5 2 0

0 0 0 0	src	dst	Parallel transfers						
0 0 0 0	src	dst	0 0 0 0 0	scde	dcde	0 0 0	0 0 0		

Description The $2s$ complement of *src* is decremented by 1 if the carry bit is set, and the result is loaded into *dst*.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a borrow occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

NOP

NOP

Syntax NOP

Operation No operation

Operands

Encoding	None	31	27	16	11	0
		0 1 1 0	x x x x x x x x x x	0 0 0 0 0	x x x x x x x x x x	x x x x x x x x x x

Description No operation. It is actually coded as a conditional transfer, but where the transfer is the operand qualifier for a data unit. Only the indicated bits are decoded, the rest are don't care.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

NOR

NOR

Syntax NOR *src1 src2 dst*

Operation *src1 NOR src2 → dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

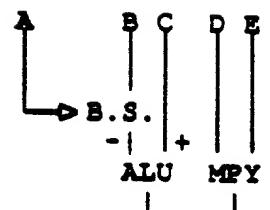
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,*src1* No parallel transfers

Routing

A	B	C	D	E
—	0	src2	src1	—



Encoding

3130		27		22		19		1615		11		8		5		2		0	
Parallel transfers																			
1	src2	src1	dst													
1	src2	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0	0	
0	1	1	1	1	...	src1	dst	H	16 bit immediate										
0	0	1	1	1	...	code	dst	H	16 bit immediate										

Description The NOR of *src1* and *src2* is loaded into *dst*. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

NOT

NOT

Syntax NOT *src,dst*

Operation (NOT *src*) \rightarrow *dst*

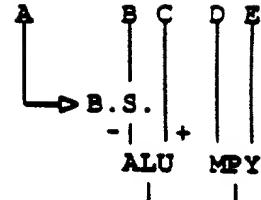
Operands

D,D With parallel transfers

any,any No parallel transfers

Routing

A	B	C	D	E
—	—	—	—	—
0	src	0	-	-



Encoding

31	27	22	19	16	11	8	5	2	0									
0	0	0	0	...	src	dst	Parallel transfers											
0	0	0	0	...	src	dst	0	0	0	0	scda	dcda	0	0	0	0	0	0

Description The 1s complement of *src* is loaded into *dst*.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Number of 1s

NUMI

Syntax NUMI src,dst

Operation the number of 1s in src \rightarrow dst

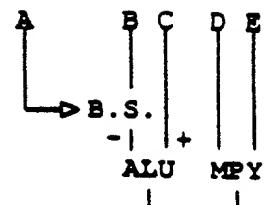
Operands

D,D With parallel transfers

any,any No parallel transfers

Routing

A	B	C	D	E
0	src	0	-	-



Encoding 31 27 22 19 16 11 8 5 2 0

0 0 0 0	src	dst	Parallel transfers					
0 0 0 0	src	dst	0	0	0	0	0	scde dcde 0 0 0 0 0 0

Description The number of 1s within src is counted and the result is loaded into dst.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - 1 if src contained all zeros, 0 otherwise.

M Bits Unaffected

Examples

OR

OR

Syntax OR *src1,src2,dst*

Operation *src1 OR src2 → dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

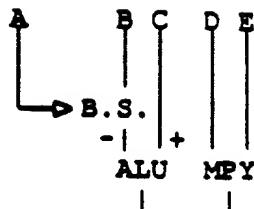
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,*src1* No parallel transfers

Routing

A	B	C	D	E
0	<i>src2</i>	<i>src1</i>	-	-



	31	30	27	22	19	16	15	11	8	5	2	0				
1	<i>src2</i>		<i>src1</i>	dst	Parallel transfers										
1	<i>src2</i>		<i>src1</i>	dst	0	0	0	0	0	<i>scde1</i>	<i>dcde</i>	<i>scde2</i>	0	0	0
0	1	1	1	<i>src1</i>	dst	H	16 bit immediate								
0	0	1	1	code	dst	H	16 bit immediate								

Description The OR of *src1* and *src2* is loaded into *dst*. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

OR with Complement

ORN

Syntax ORN src1,src2,dst

Operation src1 OR (NOT src2) \rightarrow dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

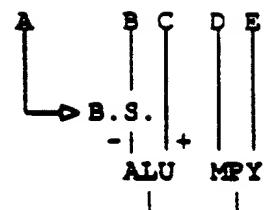
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

<u>A</u>	<u>B</u>	<u>C</u>	<u>D</u>	<u>E</u>
0	src2	src1	-	-



	31	30	27	22	19	16	15	11	8	5	2	0					
1	src2	.	.	.	src1	dst	Parallel transfers										
1	src2	.	.	.	src1	dst	0	0	0	0	scde1	dcde	scde2	0	0	0	
0 1 1 1	src1	dst	H	16 bit immediate									
0 0 1 1	code	dst	H	16 bit immediate									

Description The OR of src1 with the 1s complement of src2 is loaded into dst. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Conditional PO

POPcond

Syntax POPcond dst

Operation *A7(1+m) conditionally → dst

Operands

cond <Mn,Qn,Cn,Ln> No parallel Data Unit operation.

Encoding 31 27 232221 18 1615 1110 8 5 2 0

0	1	1	0	cond.	0	N	0	0	0	0	0	1	0	0	0	0	cde	1	1	1	dst	0	0	1
---	---	---	---	-------	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---	---	-----	---	---	---

Description This instruction will conditionally POP from the stack into a Mn, Qn, Cn or Ln register. (Note that the assembler may support POPs to the other registers which can be supported with the normal LDcond instruction). The stack pointer A7 is post-incremented. Note that the Stack Pointer is unconditionally modified. Only the register load is conditional. In addition to the normal condition codes the condition "never" is also available. This allows the Stack Pointer to be incremented without actually popping into a register.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Push Return Address

PRET

Syntax PRET

Operation (I RET {s.} *A7(-1m) conditionally)

Operands

none No parallel Data Unit operation.

Encoding 31 27 23 16 11 8 5 2 0

0	1	1	0	uncond.	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	0	1	1	0	0	1
---	---	---	---	---------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Description The value within the RET register is pushed onto the stack if the PC was loaded by either of the two previous instructions. This allows conditional calls to be supported. Theoretically this opcode format allows conditional conditional pushes of RET, but this won't be supported by the assembler.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional PUSH

PUSHcond

Syntax PUSHcond *src*

Operation *src conditionally* $\rightarrow *A7(-1m)$

Operands

cond <Mn,Qn,Cn,Ln> No parallel Data Unit operation.

Encoding 31 27 232221 18 1615 1110 8 5 2 0

0 1 1 0	cond.	0 N 0 0 0 0 0 0	1 0 0 0 1	cde	1 1 1	dst	0 0 1
---------	-------	-----------------	-----------	-----	-------	-----	-------

Description This instruction will conditionally PUSH from a Mn, Qn, Cn or Ln register to the stack. (Note that the assembler may support PUSHes of the other registers which can be supported with the normal STcond instruction). The stack pointer A7 is pre-decremented. Note however that the Stack Pointer is unconditionally modified. Only the register store is conditional. In addition to the normal condition codes the condition "never" is also available. This allows the Stack Pointer to be incremented without actually pushing anything onto the stack.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Right-most 1

R.M1

Syntax R.M1 src,dst

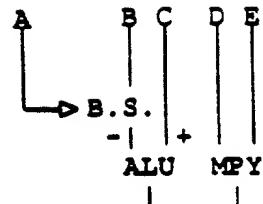
Operation bit number of right-most 1 in src \rightarrow dst

Operands D,D With parallel transfers

any,any No parallel transfers

Routing

A	B	C	D	E
0	src	0	-	-



Encoding	3130	27	22	19	16	11	8	5	2	0
Parallel transfers										
	0 0 0 0	src	dst			0 0 0 0 0	scde	dcde	0 0 0 0 0 0

Description The bit number of the right-most 1 in src is loaded into dst.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - 1 if src contained all zeros, 0 otherwise.

M Bits Unaffected

Examples

Rotate

ROT

Syntax ROT *src1,src2,dst*

Operation *src1* rotated left by amount in *src2* → *dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

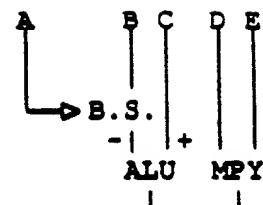
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,*src1* No parallel transfers

Routing

A	B	C	D	E
src2	src1	0	-	-



Encoding 3130 27 22 19 1615 11 8 5 2 0

Parallel transfers									
1	src2	src1	dst					
1	src2	src1	dst	0 0 0 0 0	scde1	dcde	scde2	0 0 0
0 1 1 1	src1	dst	H	16 bit immediate				
0 0 1 1	code	dst	H	16 bit immediate				

Description *src1* is left-rotated by the amount in the least-significant 5 bits of *src2*. The result is loaded into *dst*. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - Unaffected

C - Set to the last value rotated out. 0 if rotated by 0.

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Shift and Add

SADD

Syntax SADD *src1,src2,dst*

Operation *src1 + (src2 shifted by pre-defined amount) → dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

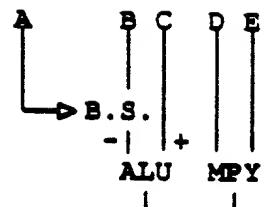
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,*src1* No parallel transfers

Routing

A	B	C	D	E
—	pre	src2	src1	—



Encoding

3130		27	22	19	1615	11	8	5	2	0
1	src2	...	src1	dst	Parallel transfers					
1	src2	...	src1	dst	0 0 0 0 0	scde1	dcde	scde2	0 0 0	
0 1 1 1	...	src1	dst	H	16 bit immediate					
0 0 1 1	...	code	dst	H	16 bit immediate					

Description *src2* is shifted by the pre-defined shift amount in the OPTIONS register. This is then added to *src1* and the result loaded into *dst*. The immediate is assumed to be unsigned, and before shifting, can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a carry occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Set Bit

vF

Syntax SETB *src1.src2.dst*

Operation *src1[src2]:=1, src1' → dst*

Operands

D,D,D With parallel transfers

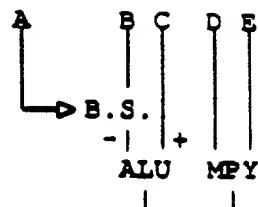
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing



Encoding																	
31	30	27	22	19	16	15	11	8	5	2	0						
1	src2	src1	dst	Parallel transfers											
1	src2	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0	
0	1	1	1	...	src1	dst	H	16 bit immediate									
0	0	1	1	...	code	dst	H	16 bit immediate									

Description The bit within *src1* pointed to by the least-significant 5-bits of *src2* is set to one. The result is loaded into *dst*. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - Set to 0?

M Bits Unaffected

Examples

Shift Left Logica

Syntax SL src1,src2,dst

Operation src1 shifted left by src2, 0 fill \rightarrow dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

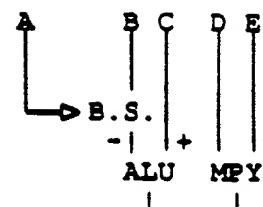
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
src2	src1	0	-	-



Encoding 31 30 27 22 19 16 15 11 8 5 2 0

	src2	...	src1	dst	Parallel transfers								
1	src2	...	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0 0 0
0 1 1 1	...	src1	dst	H	16 bit immediate								
0 0 1 1	...	code	dst	H	16 bit immediate								

Description src1 is shifted left by the amount indicated in the least-significant 5 bits of src2. The least-significant bits are zero filled. The result is loaded into dst. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - Unaffected

C - Set to the last value shifted out. 0 if shift amount was 0.

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Shift Right Arithmetic

SR A

Syntax SRA *src1,src2,dst*

Operation *src1* shifted right by *src2*, sign extended \rightarrow *dst*

Operands

D,D,D With parallel transfers

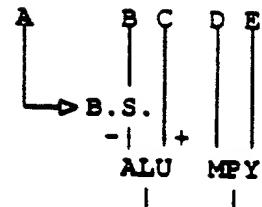
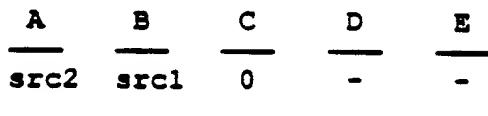
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,*src1* No parallel transfers

Routing



	31	30	27	22	19	16	15	11	8	5	2	0			
1	<i>src2</i>	<i>src1</i>	<i>dst</i>	Parallel transfers										
1	<i>src2</i>	<i>src1</i>	<i>dst</i>	0	0	0	0	0	<i>scde1</i>	<i>dcde</i>	<i>scde2</i>	0	0	0
0 1 1 1	<i>src1</i>	<i>dst</i>	H	16 bit immediate										
0 0 1 1	<i>code</i>	<i>dst</i>	H	16 bit immediate										

Description *src1* is shifted right by the amount indicated in the least-significant 5 bits of *src2*. The sign-bit is copied into the most-significant bits. The result is loaded into *dst*. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - Set to the last value shifted out. 0 if shift amount was 0.

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Shift Right Logic

Syntax **SRL** *src1,src2,dst*

Operation *src1* shifted right by *src2*, 0 fill \rightarrow *dst*

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

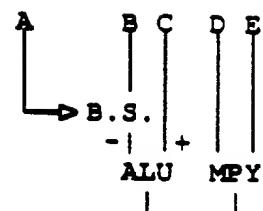
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
<u>src2</u>	<u>src1</u>	0	-	-



Encoding									
		31	30	27	22	19	16	15	11
1	src2		src1	dst	Parallel transfers			
1	src2		src1	dst	0	0	0	scde1 dcde scde2 0 0 0
0 1 1 1		src1	dst	H	16 bit immediate		
0 0 1 1		code	dst	H	16 bit immediate		

Description *src1* is shifted right by the amount indicated in the least-significant 5 bits of *src2*. The most-significant bits are zero filled. The result is loaded into *dst*. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - Unaffected

C - Set to the last value shifted out. 0 if shift amount was 0.

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Shift and Subtract

Syntax SSUB src1.src2.dst

Operation src1 - (src2 shifted by pre-defined amount) → dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

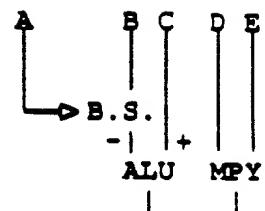
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
—	pre	src2	src1	—



	31	30	27	22	19	16	15	11	8	5	2	0				
Parallel transfers																
1	src2	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0
1	src2	src1	dst	H										
0 1 1 1	src1	dst	H										
0 0 1 1	code	dst	H										

Description src2 is shifted by the pre-defined shift amount in the OPTIONS register. This is then subtracted from src1 and the result loaded into dst. The immediate is assumed to be unsigned, and before shifting, can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a carry occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Conditional store

Syntax STcond dst,*An(mode)

Operation src conditionally $\rightarrow *dst$

Operands

cond A0-A7(mode{g},<Dn.An.Xn.Pn>) No parallel Data Unit operation.

Addressing modes pre- or post-indexing

+/- 3 bit immediate or +/- an index register

modify Address register, or leave unaltered

Encoding	31	27	232221	18	1615	1110	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 0 0 0	mod	1 cde	A	dst	Imm/X		

Description This instruction will conditionally store a Dn, An, Xn or Pn register to an indirect address generated from any Address register. (Mn, Qn, Cn or Ln cannot be directly stored to memory except with a PUSH instruction). If specifying an immediate then +/- 0 to 7 are available. If specifying an index register then it must be in the same Address sub-unit as the Address register. If the address is non-aligned, then this will store only the lower byte(s). Note that if a register modify is specified then this will happen unconditionally. Only the actual store is conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modify but without storing a register.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional Store / Load

Syntax STcond *src1,*Am(mode)* || LD **An(mode),dst2*

Operation (I *src1* conditionally (s.) **dst1*, **src2* conditionally → *dst2*)

Operands

cond Dm.A0-A3(mode{g} A4-A7(mode),Dn) No parallel Data Unit operation.

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated indeX register with Address register modify

indirect without indexing

Encoding	31	27	232221	18	1615	1110	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 0 0 1	mdg	mdl	1 A47	0 A03	Dg	D1	

Description This instruction will conditionally store *src1* via the Global bus to an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will conditionally (same condition) load *dst2* via the Local bus from an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independantly on the two buses. The indeX register(s) used have the same subscript(s) as the Address register(s). *src1* and *dst2* must be D registers. Note that if register modifies are specified then these will happen unconditionally. Only the actual store and load are conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modifies but without loading or storing anything.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Conditional Store - STcnd

Syntax STcond *src1,*Am(mode)* || ST *src2,*An(mode)*

Operation {I *src1* conditionally {s.} **dst1*, *src2* conditionally {s.} **dst2*}

Operands

cond Dm,A0-A3(mode{g}) Dn,A4-A7(mode)) No parallel Data Unit operation.

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated index register with Address register modify

indirect without indexing

Encoding	31	27	232221	18	1615	1110	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0 0 0 1	mdg	mdl	1 A47	1 A03	Dg	D1	

Description This instruction will conditionally store *src1* via the Global bus to an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will store *src2* via the Local bus to an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independently on the two buses. The index register(s) used have the same subscript(s) as the Address register(s). *src1* and *src2* must be D registers. Note that if register modifies are specified then these will happen unconditionally. Only the actual stores are conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modifies but without storing anything.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Store Upper Conditionally

STUcond

Syntax STUcond *src,*An(mode)*

Operation *src* → **dst*

Operands

<Dn,An,Xn,Pn>,A0-A7(mode(g)) No parallel Data Unit operation.

Addressing modes pre- or post-indexing .

+/- 3 bit immediate or +/- an index register

modify Address register, or leave unaltered

Encoding	31	27	232221	18	1615	1110	8	5	2	0
	0 1 1 0	cond.	0 N 0 0 0	1 0 0	mod	1 cde	A	dst	Imm/X	

Description For use with non-aligned addresses. Conditionally stores the upper portion of a PP register to an indirect non-aligned address generated from any Address register. The source register must be one of Dn, An, Xn or Pn. (Non-aligned stores are not supported for any other PP registers). Note that if register modify is specified then this will happen unconditionally. Only the actual store is conditional. In addition to the normal condition codes the condition "never" is also available. This allows Address register modify but without loading a register. If specifying an immediate then +/- 0 to 7 are available. If specifying an index register then it must be in the same Address sub-unit as the Address register.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax SUB src1,src2,dst

Operation src1 - src2 → dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

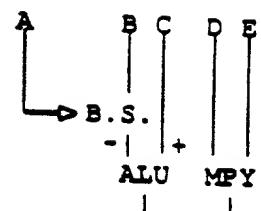
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
—	src2	src1	-	-



Encoding	3130	27	22	19	1615	11	8	5	2	0
	Parallel transfers									
1	src2	src1	dst						
1	src2	src1	dst	0 0 0 0 0	scde1	dcde	scde2	0 0 0	
0 1 1 1	src1	dst	H	16 bit immediate					
0 0 1 1	code	dst	H	16 bit immediate					

Description src2 is subtracted from src1 and the result is loaded into dst. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a carry occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Syntax SUBB src1,src2,dst

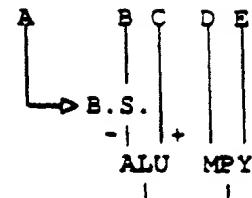
Operation src1 - src2 - C → dst

Operands

- D,D,D With parallel transfers
- any,D,any No parallel transfers
- D,any,any No parallel transfers
- D,I,D No parallel transfers
- any,I,src1 No parallel transfers

Routing

A	B	C	D	E
—	src2	src1	-	-
0				



Encoding

31	30	27	22	19	16	15	11	8	5	2	0							
1	src2	...	src1	dst	Parallel transfers													
1	src2	...	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0			
0	1	1	1	...	src1	dst	H	16 bit immediate										
0	0	1	1	...	code	dst	H	16 bit immediate										

Description (src2 + the Carry bit) is subtracted from src1 and the result is loaded into dst. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - 1 if negative result generated, 0 otherwise.

C - 1 if a borrow occurs, 0 otherwise.

V - 1 if an overflow occurs, 0 otherwise.

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Syntax SUBM src1 src2 dst

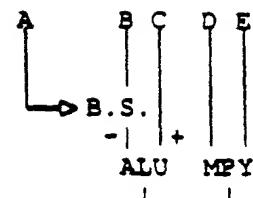
Operation src1 ----/-/- src2 → dst

Operands

- D,D,D With parallel transfers
- any,D,any No parallel transfers
- D,any,any No parallel transfers
- D,I,D No parallel transfers
- any,I,src1 No parallel transfers

Routing

A	B	C	D	E
0	src2	src1	-	-



Encoding

31	30	27	22	19	16	15	11	8	5	2	0				
Parallel transfers															
1	src2	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0
0	1	1	1	src1	dst	H			16 bit immediate					
0	0	1	1	code	dst	H			16 bit immediate					

Description The ALU bits in the OPTIONS register are used to split the ALU into four separate bytes, two separate half-words, or one word. Each portion of src2 is subtracted from the corresponding portion of src1, and the result(s) stored in dst. The individual borrows are stored into the 4, 2 or 1 least-significant M bits respectively. The immediate is assumed to be unsigned, and can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits The borrows from the individual subtractions are stored in the least-significant 4, 2 or 1 M bits, according to the ALU option bit values.

Examples

Syntax TSTB src1,src2,dst

Operandon src1[src2] is tested for 1

Operands

D,D,D With parallel transfers

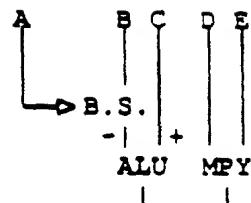
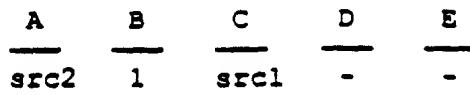
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing



Encoding 31 30 27 22 19 16 15 11 8 5 2 0

1	src2	src1	0 0 0	Parallel transfers							
1	src2	src1	0 0 0	0 0 0 0 0	scde1	0 0 0	scde2	0 0 0			
0 1 1 1	src1	0 0 0	H	16 bit immediate							
0 0 1 1	code	src1	H	16 bit immediate							

Description The bit within src1 pointed to by the least-significant 5-bits of src2 is tested for one. The Z status bit is set to 1 if the bit was zero, 0 otherwise. Only the least-significant 5 bits of the immediate are significant.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if bit was zero, 0 otherwise

M Bits Unaffected

Examples

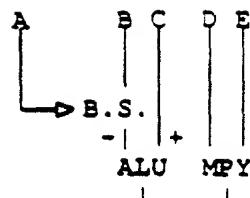
Syntax ULCK

Operanon unlock MIMD PPs from each other

Operands None With parallel transfers

Routing

A B C D E
— — — — —



Encoding 31 27 22 19 16 0
0 0 0 0 | | 0 0 0 0 0 0 | Parallel transfers

Description Instruction unlocks the MIMD PPs from each other. They then resume independant instruction execution on the next instruction fetch.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax XNOR src1,src2,dst

Operation src1 XNOR src2 → dst

Operands

D,D,D With parallel transfers

any,D,any No parallel transfers

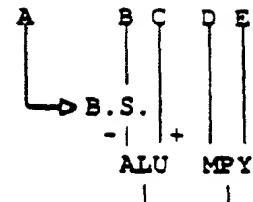
D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing

A	B	C	D	E
0	src2	src1	-	-



Encoding 31 30 27 22 19 16 15 11 8 5 2 0

1	src2	src1	dst	Parallel transfers										
1	src2	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0
0	1	1	1	src1	dst	H	16 bit immediate							
0	0	1	1	code	dst	H	16 bit immediate							

Description The exclusive NOR of src1 and src2 is loaded into dst. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Syntax XOR src1.src2.dst

Operation src1 XOR src2 → dst

Operands

D,D,D With parallel transfers

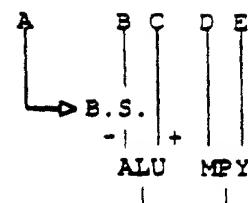
any,D,any No parallel transfers

D,any,any No parallel transfers

D,I,D No parallel transfers

any,I,src1 No parallel transfers

Routing



Encoding 3130 27 22 19 1615 11 8 5 2 0

1	src2	...	src1	dst	Parallel transfers										
1	src2	...	src1	dst	0	0	0	0	0	scde1	dcde	scde2	0	0	0
0	1	1	1	...	src1	dst	H	16 bit immediate							
0	0	1	1	...	code	dst	H	16 bit immediate							

Description The exclusive OR of src1 and src2 is loaded into dst. The immediate can be either the high or low half-word.

Status Bits N - Unaffected

C - Unaffected

V - Unaffected

Z - 1 if zero result generated, 0 otherwise.

M Bits Unaffected

Examples

Syntax || LD *An(mode).dst

Operan *src → dst

Operands

A0-A7(mode{g })<Dn.An.Xn.Pn>) With parallel Data Unit operation

Addressing modes pre- or post-indexing

+/- 3 bit immediate or +/- an index register

modify Address register, or leave unaltered

Encoding 31 27 232221 18 1615 1110 8 5 2 ?

Data Unit operation		0	mod	0	cde	A	dst	Imm/X
---------------------	--	---	-----	---	-----	---	-----	-------

Description In parallel with a (D register) Data Unit operation will load a Dn, An, Xn or Pn register from an indirect address generated from any Address register. (Mn, Qn, Cn or Ln cannot be directly loaded from memory except with a ||POP instruction). If specifying an immediate then +/- 0 to 7 are available. If specifying an index register then it must be in the same Address sub-unit as the Address register. If the address is non-aligned, then this will load only the lower byte(s).

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax || LD *Am(mode).dst1 || LD *An(mode).dst2

Operation *src1 → dst1 || *src2 → dst2

Operands

A0-A3(mode(g)) .Dm A4-A7(mode),Dn) With parallel Data Unit operation,
or

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated index register with
Address register modify

indirect without indexing

<i>Encoding</i>	31	27	232221	18	1615	1110	8	7	5	2	0
			Data Unit operation		1 mdg md1 0 A47 0 A03 Dg D1						

Description In parallel with a (D register) Data Unit operation, will load dst1 via the Global bus from an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will load dst2 via the Local bus from an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independently on the two buses. The index register(s) used have the same subscript(s) as the Address register(s). The dsts must be D registers.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax || LD *Am(mode).dst1 || ST src2.*An(mode)

Operanon *src1 → dst1 || src2 → *dst2

Operands

A0-A3(mode{g}).Dm Dn,A4-A7(mode)) With parallel Data Unit operation.
or

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated indeX register with
Address register modify

indirect without indexing

Encoding 31 27 232221 18 1615 1110 2 7 5 2 0

Data Unit operation		1	mdg	mdl	0	A47	1	A03	Dg	D1
---------------------	--	---	-----	-----	---	-----	---	-----	----	----

Description In parallel with a (D register) Data Unit operation, will load dst1 via the Global bus from an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will store src2 via the Local bus to an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independantly on the two buses. The indeX register(s) used have the same subscript(s) as the Address register(s). dst1 and src2 must be D registers.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax || MOV src.dst;

Operation src → dst

Operands

any,any With parallel Data Unit operation

<i>Encoding</i>	31	27	232221	16	11	8	5	2	0
Data Unit Operation									

Description In parallel with a (D register) Data Unit operation, will move any register src to any other register dst.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax // POP dst

Operation *A7(I+m) → dst

Operands

<Mn,Qn,Cn,Ln> With parallel Data Unit operation

<i>Encoding</i>	31	27	232221	18	1615	1110	8	5	2	0
			Data Unit operation	0	1 0 0 0 0	cde	1 1 1	dst	0 0 1	

Description In parallel with a (D register) Data Unit operation, will POP from the stack into a Mn, Qn, Cn or Ln register. (Note that the assembler may support //POPs to the other registers which can be supported with the normal //LD instruction). The stack pointer A7 is post-incremented.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax **||PRET**

Operanon **(I RET (s.) *A7(-1m) conditionally)**

Operands

none With parallel Data Unit operation.

Encoding

31	16	11	8	5	2	0
	Data Unit Operation	0 0 0 1 0	0 0 1	1 1 1	0 1 1	0 0 1

Description In parallel with a (D register) Data Unit operation, the value within the RET register is pushed onto the stack if the PC was loaded by either of the two previous instructions. This allows conditional calls to be supported.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax || PUSH src

Operation *src* → *A7(-Im)

Operands

< Mn, Qn, Cn, Ln > With parallel Data Unit operation

Encoding

Data Unit operation	0	1	0	0	0	1	cde	1	1	1	dst	0	0	1
---------------------	---	---	---	---	---	---	-----	---	---	---	-----	---	---	---

Description In parallel with a (D register) Data Unit operation, will PUSH from a Mn, Qn, Cn or Ln register to the stack. (Note that the assembler may support !!PUSHs of the other registers which can be supported with the normal !!ST instruction). The stack pointer A7 is pre-decremented.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

Syntax $\text{II ST } \text{src}.\ast\text{An(mode)}$

Operation $\text{src} \rightarrow \ast\text{dst}$

Operands

A0-A7(mode{g },<Dn.An.Xn.Pn>) With parallel Data Unit operation

Addressing modes pre- or post-indexing

+/- 3 bit immediate or +/- an index register

modify Address register, or leave unaltered

<i>Encoding</i>	31	27	232221	18	1615	1110	8	5	2	0
Data Unit operation										

Description In parallel with a (D register) Data Unit operation will store a Dn, An, Xn or Pn register to an indirect address generated from any Address register. (Mn, Qn, Cn or Ln cannot be directly stored to memory except with a II PUSH instruction). If specifying an immediate then +/- 0 to 7 are available. If specifying an index register then it must be in the same Address sub-unit as the Address register. If the address is non-aligned, then this will store only the lower byte(s).

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bus Unaffected

Examples

Syntax || ST *src1,*Am(mode)* || LD **An(mode).dst2*

Operation *src1* → **dst1* || **src2* → *dst2*

Operands

Dm,A0-A3(mode|g) A4-A7(mode),Dn) With parallel Data Unit operation, or

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated indeX register with
Address register modify

indirect without indexing

Encoding 31 27 232221 18 1615 1110 8 5 2 0

Data Unit operation	1	mdg	mdl	1	A47	0	A03	Dg	D1
---------------------	---	-----	-----	---	-----	---	-----	----	----

Description In parallel with a (D register) Data Unit operation, will store *src1* via the Global bus to an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will load *dst2* via the Local bus from an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independantly on the two buses. The indeX register(s) used have the same subscript(s) as the Address register(s). *src1* and *dst2* must be D registers.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bus Unaffected

Examples

Syntax $\| ST \ src1.*A_m(mode) \| ST \ src2.*A_n(mode)$

Operation $src1 \rightarrow *dst1 \| src2 \rightarrow *dst2$

Operands

D_n.A0-A3(mode(g)) D_n.A4-A7(mode)) With parallel Data Unit operation, or

Addressing modes post-increment by 1 with Address register modify

pre-decrement by 1 with Address register modify

post-increment by A reg's associated index register with
Address register modify

indirect without indexing

<i>Encoding</i>	31	27	232221	18	1615	1110	8	5	2	0
			Data Unit operation		1 mdg mdl 1 A47 0 A03	Dg	D1			

Description In parallel with a (D register) Data Unit operation, will store *src1* via the Global bus to an indirect address generated from an Address register in the Global sub-unit (A0-A3). In parallel with this it will store *src2* via the Local bus to an indirect address generated from an Address register in the Local sub-unit (A4-A7). Indirect, post-increment by 1, pre-decrement by 1 and post-increment by X addressing modes are supported independently on the two buses. The index register(s) used have the same subscript(s) as the Address register(s). *src1* and *src2* must be D registers.

Status Bits N - Unaffected.

C - Unaffected.

V - Unaffected.

Z - Unaffected.

M Bits Unaffected

Examples

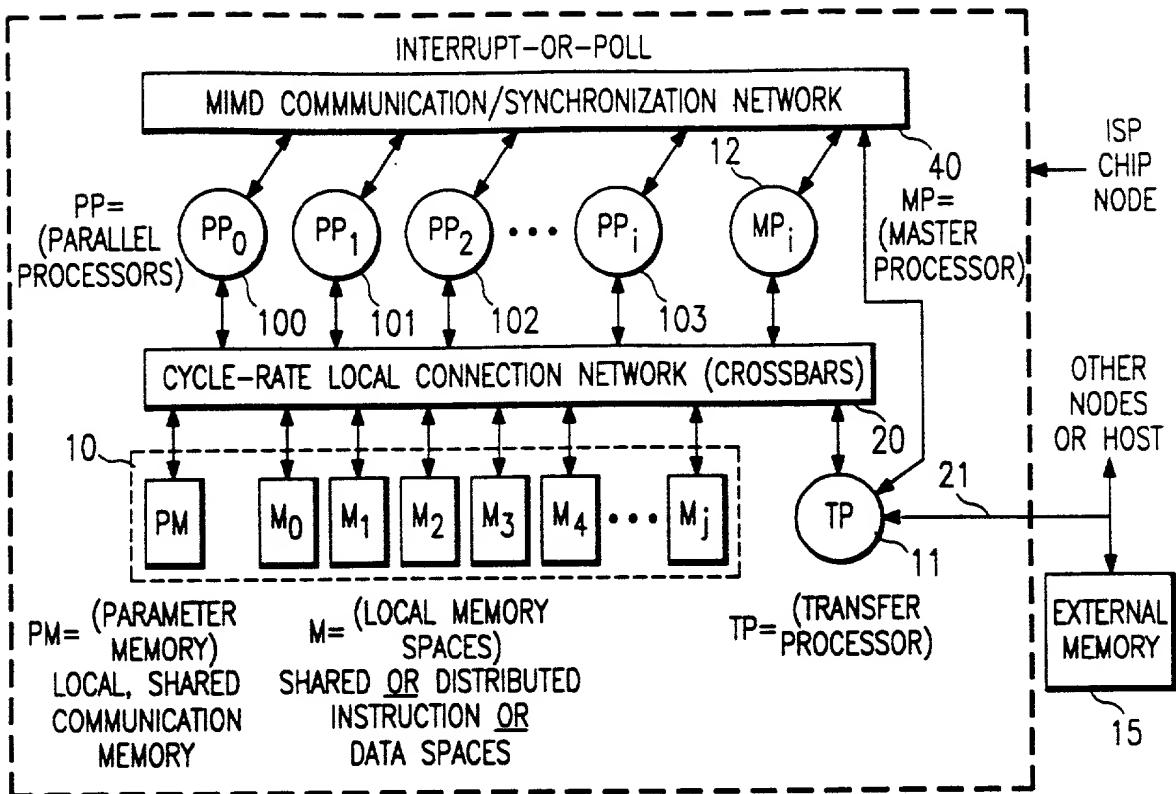
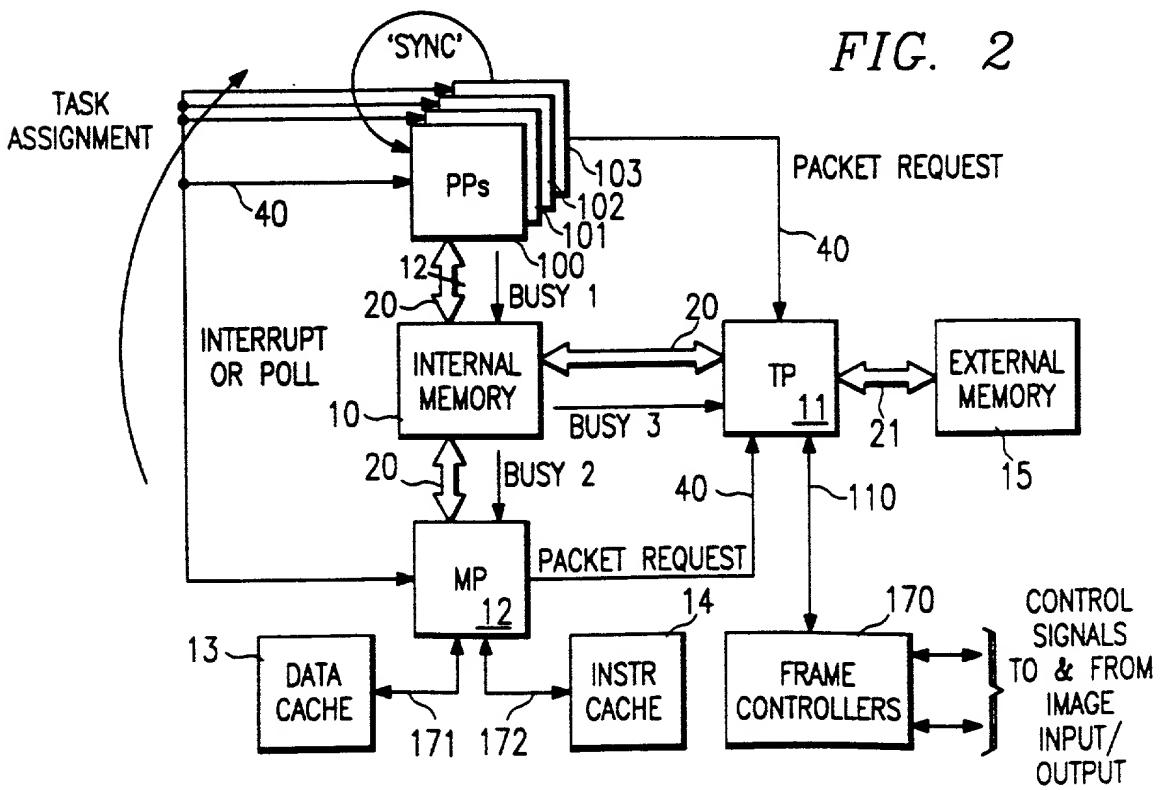


FIG. 1



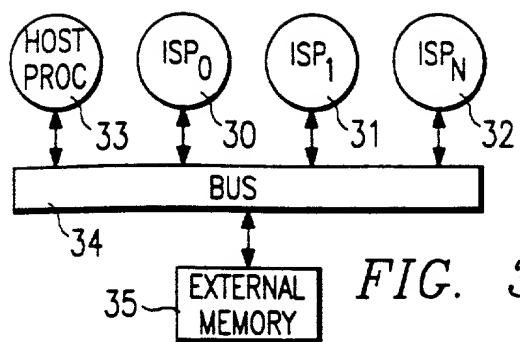
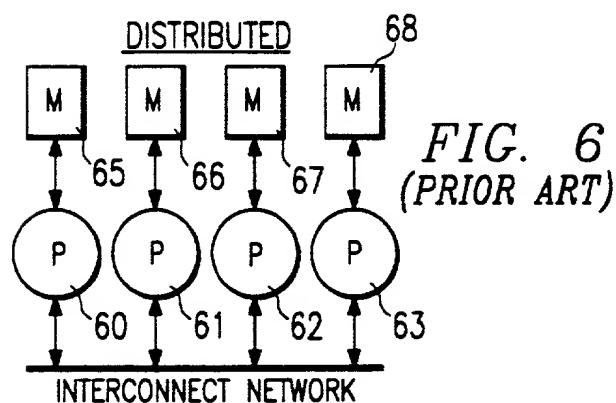


FIG. 3



*FIG. 6
(PRIOR ART)*

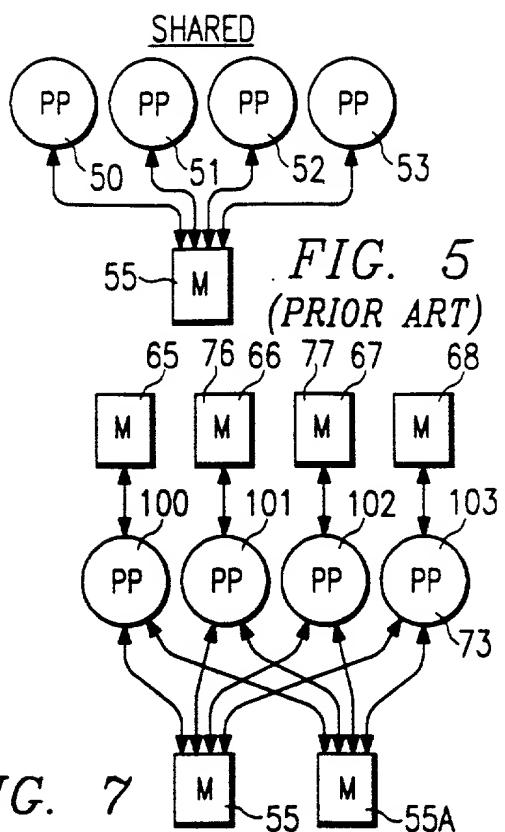
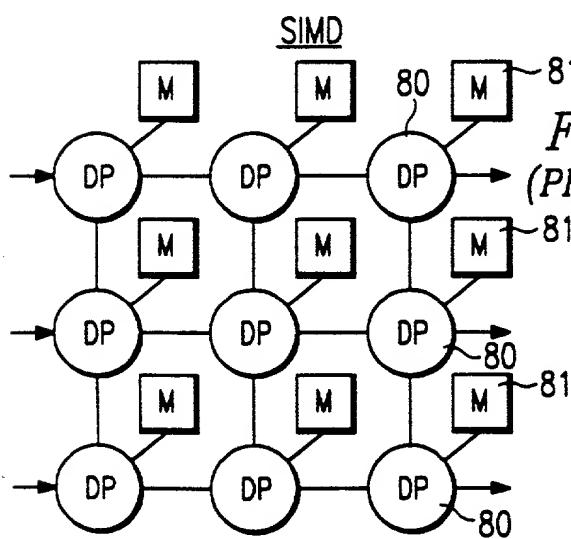
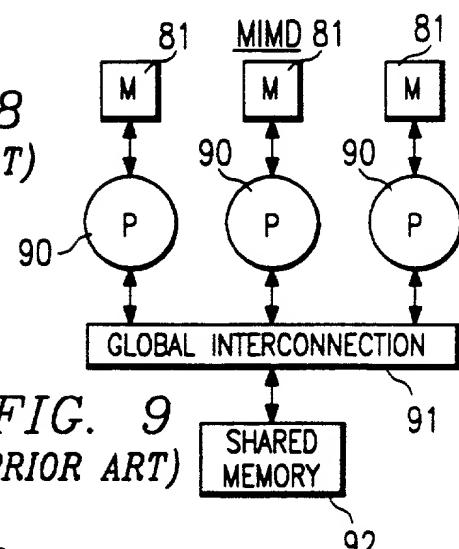


FIG. 7



*FIG. 8
(PRIOR ART)*



*FIG. 9
(PRIOR ART)*

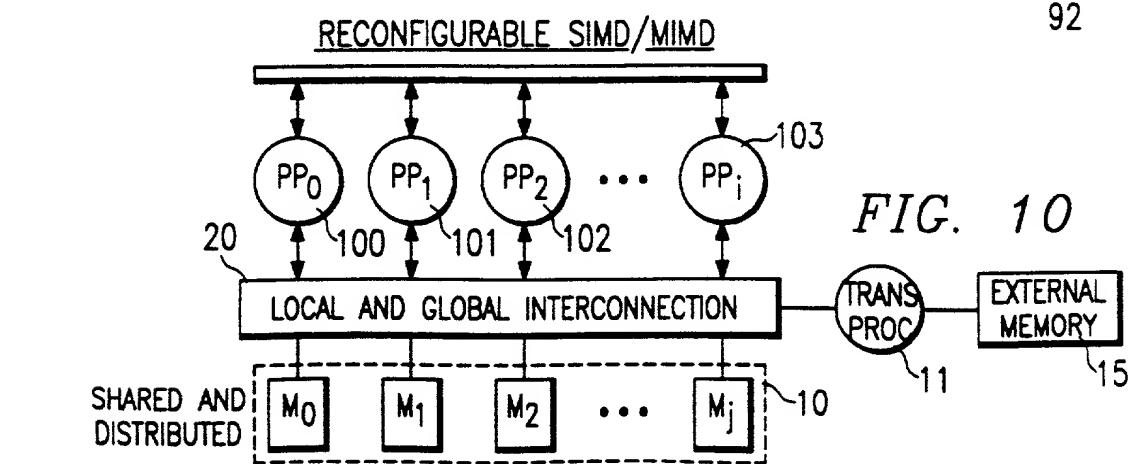


FIG. 10

FIG. 4

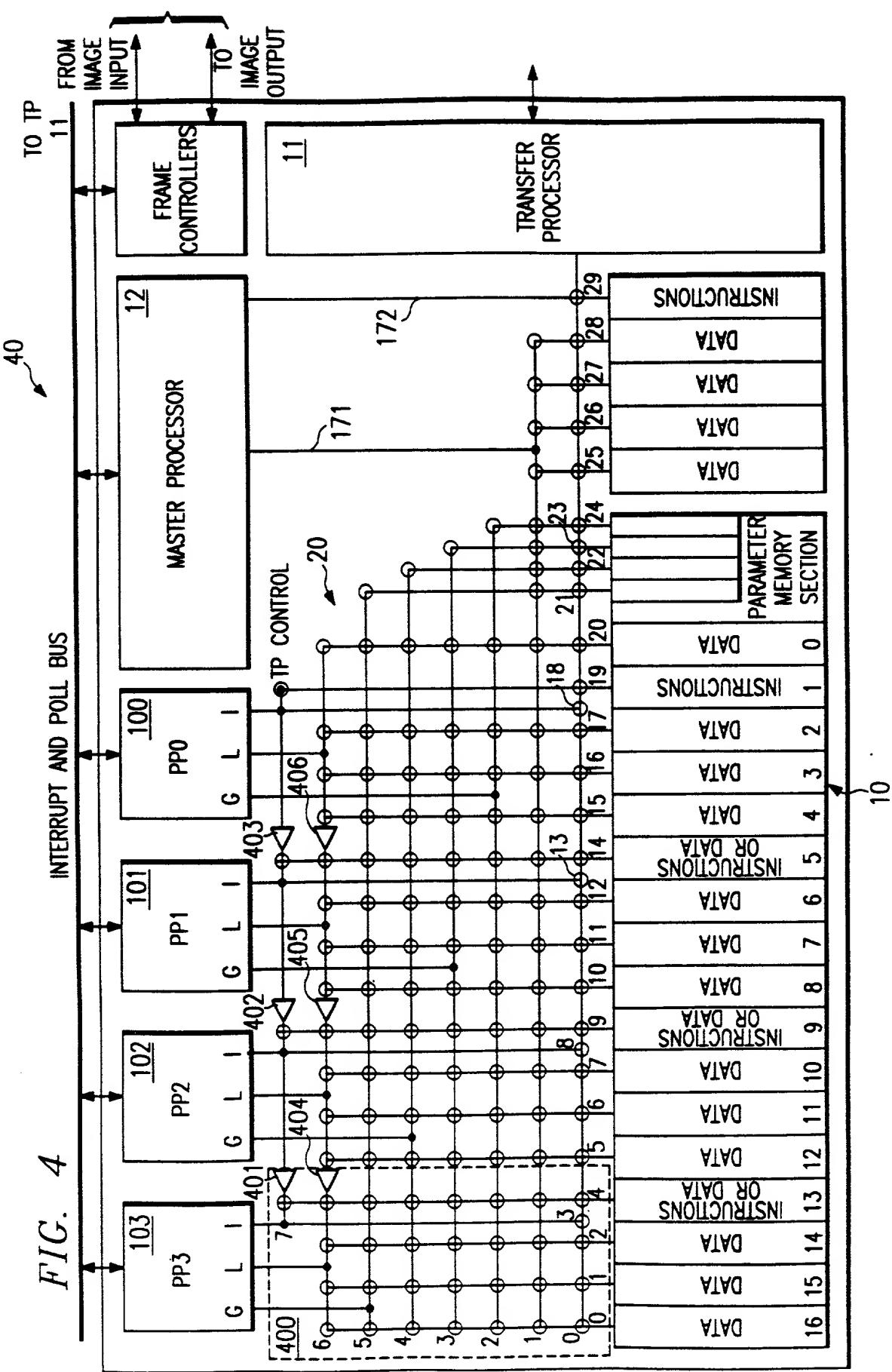


FIG. 11

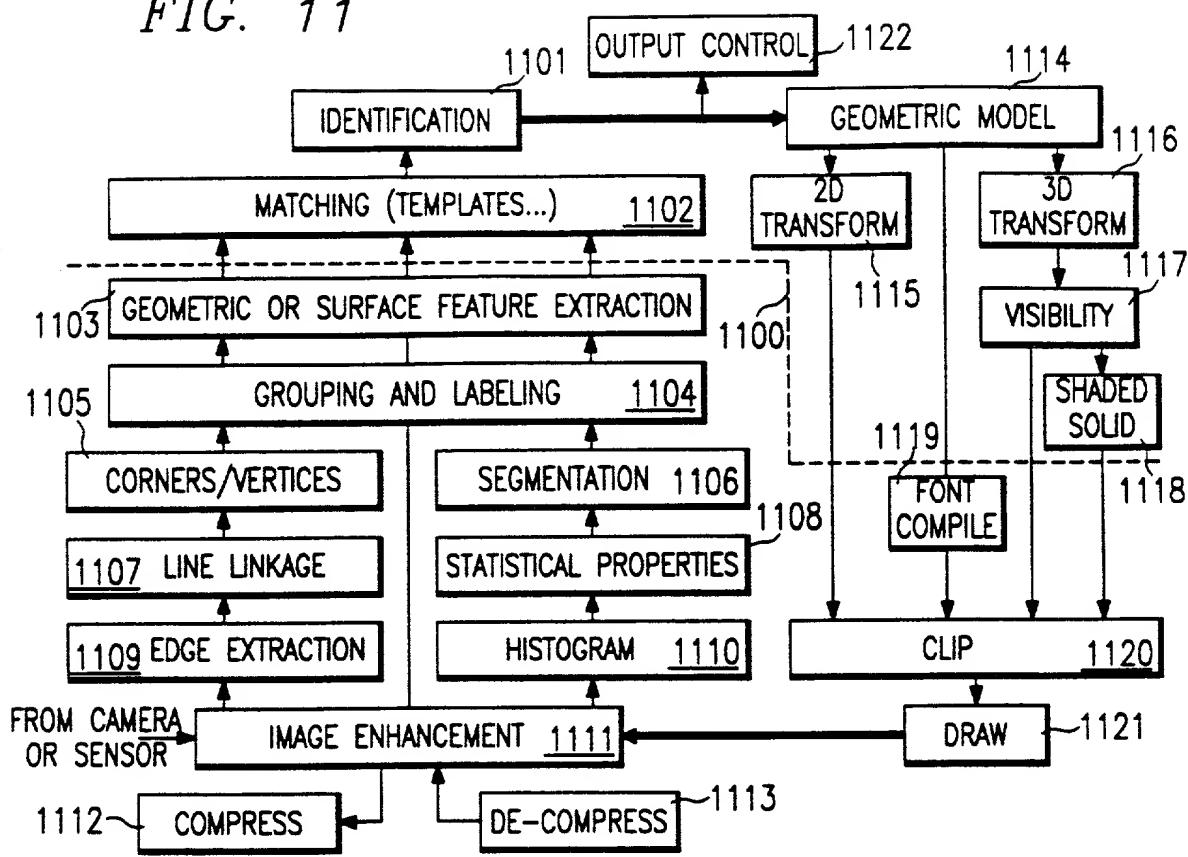


FIG. 12

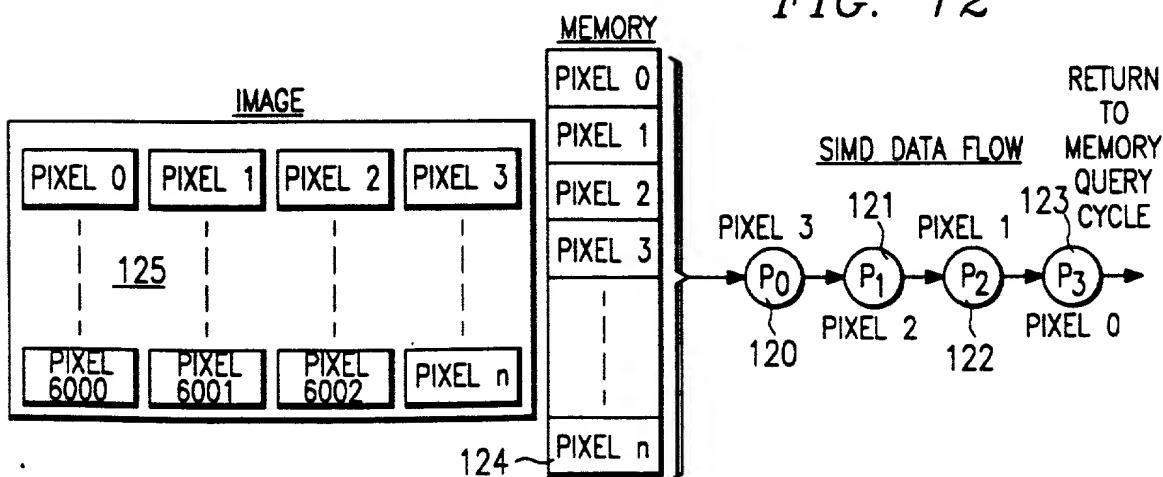


FIG. 13

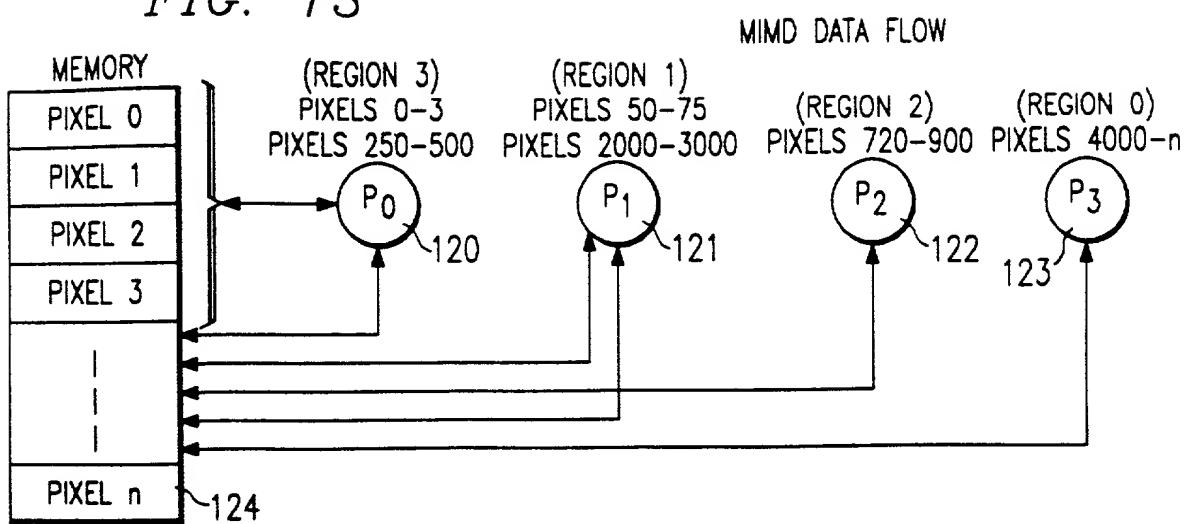


FIG. 14

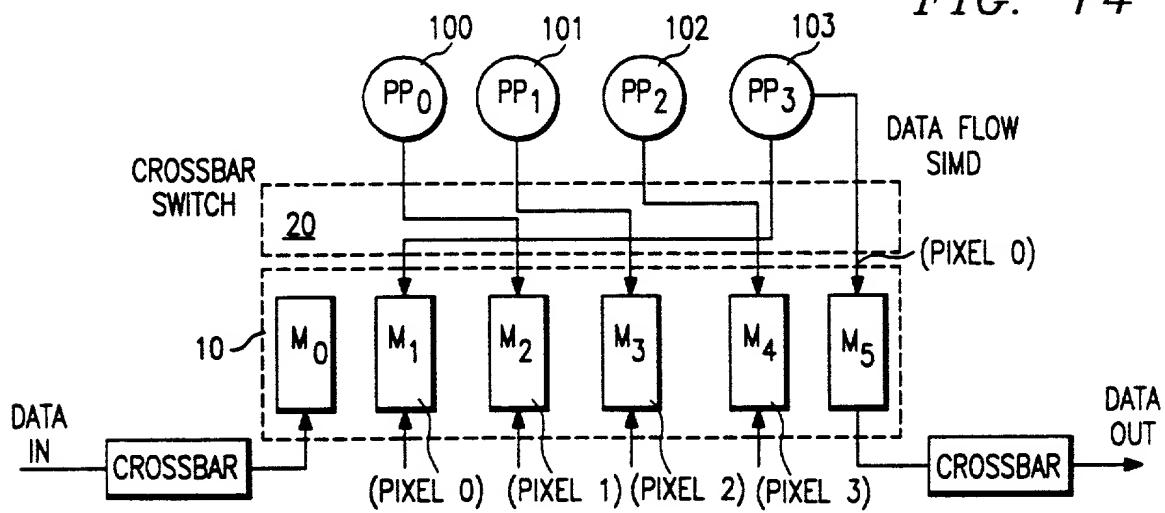
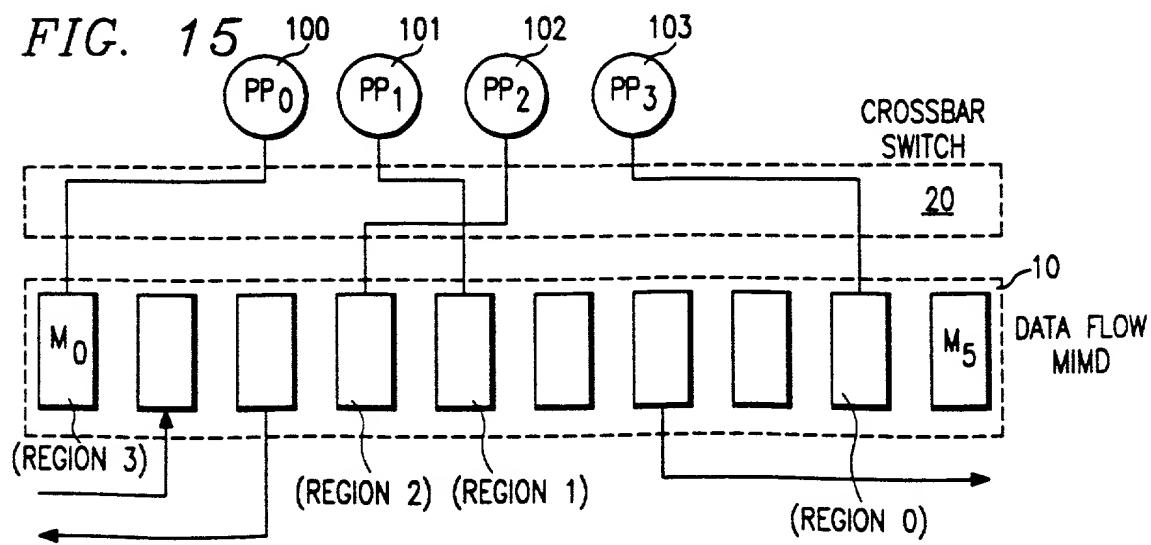


FIG. 15



100 101 102 103 104 105 106 107

FIG. 16

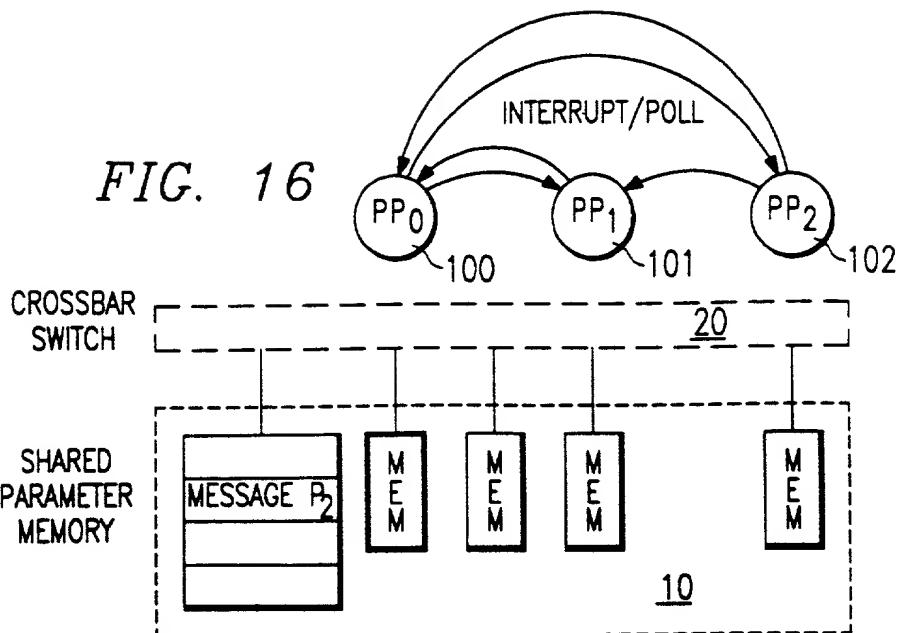


FIG. 17

